Programando na Unity

Muito bem pessoal! Como vamos desenvolver jogos em 2D é primordial que o nosso game tenha interações, mecânica de games, etc.

Para isso vamos utilizar o C# que é uma das linguagens de programação mais utilizada mundialmente e principalmente no desenvolvimento de games na Unity. Neste módulo, vamos conhecer sua estrutura e seus principais comandos que serão responsáveis pela estrutura de backend do nosso game.

Para isso, vamos já criar um arquivo de programação em C# dentro da pasta Script:

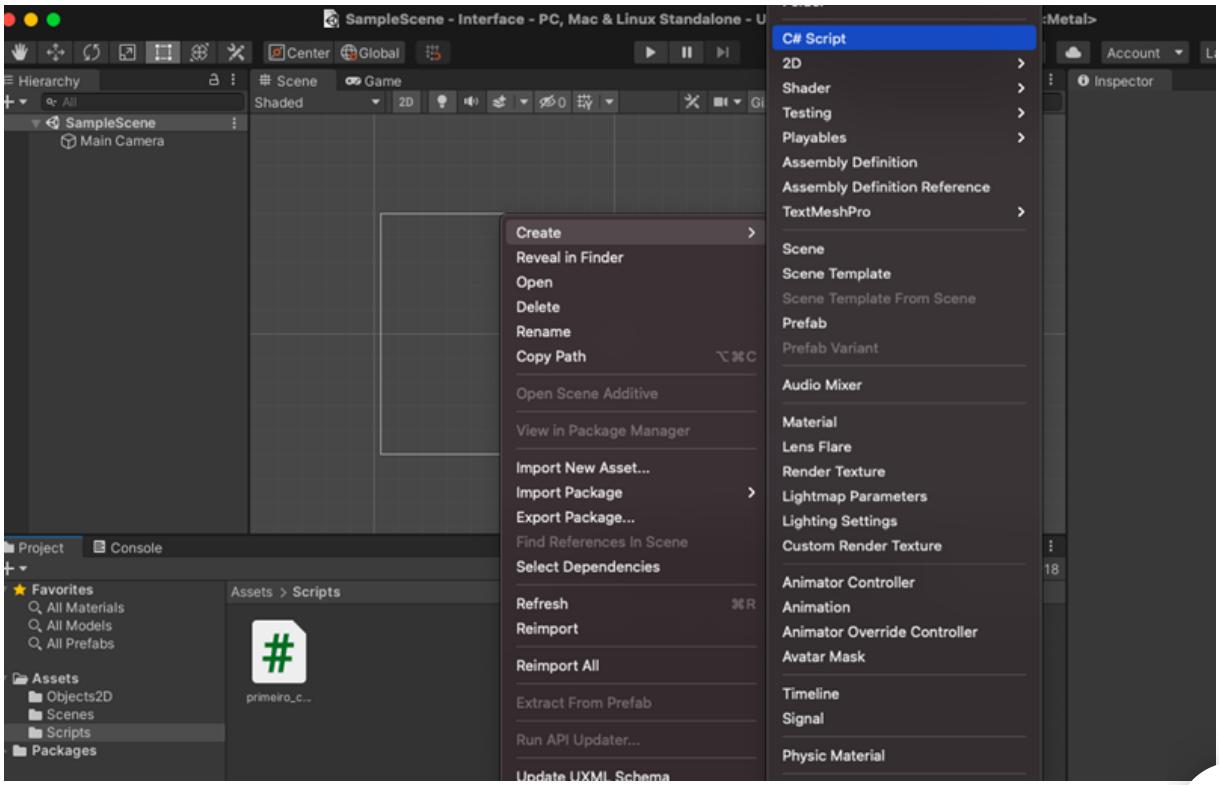

Imagem 1: Tela da Unity | Fonte: Autor (print de tela

Com este procedimento, a unity irá criar nosso primeiro script de programação em C#. Vamos renomear este script para: primeiro cód Veja que, no painel inspector, já aparece a estrutura do nosso arquivo e com alguns códigos de programação:

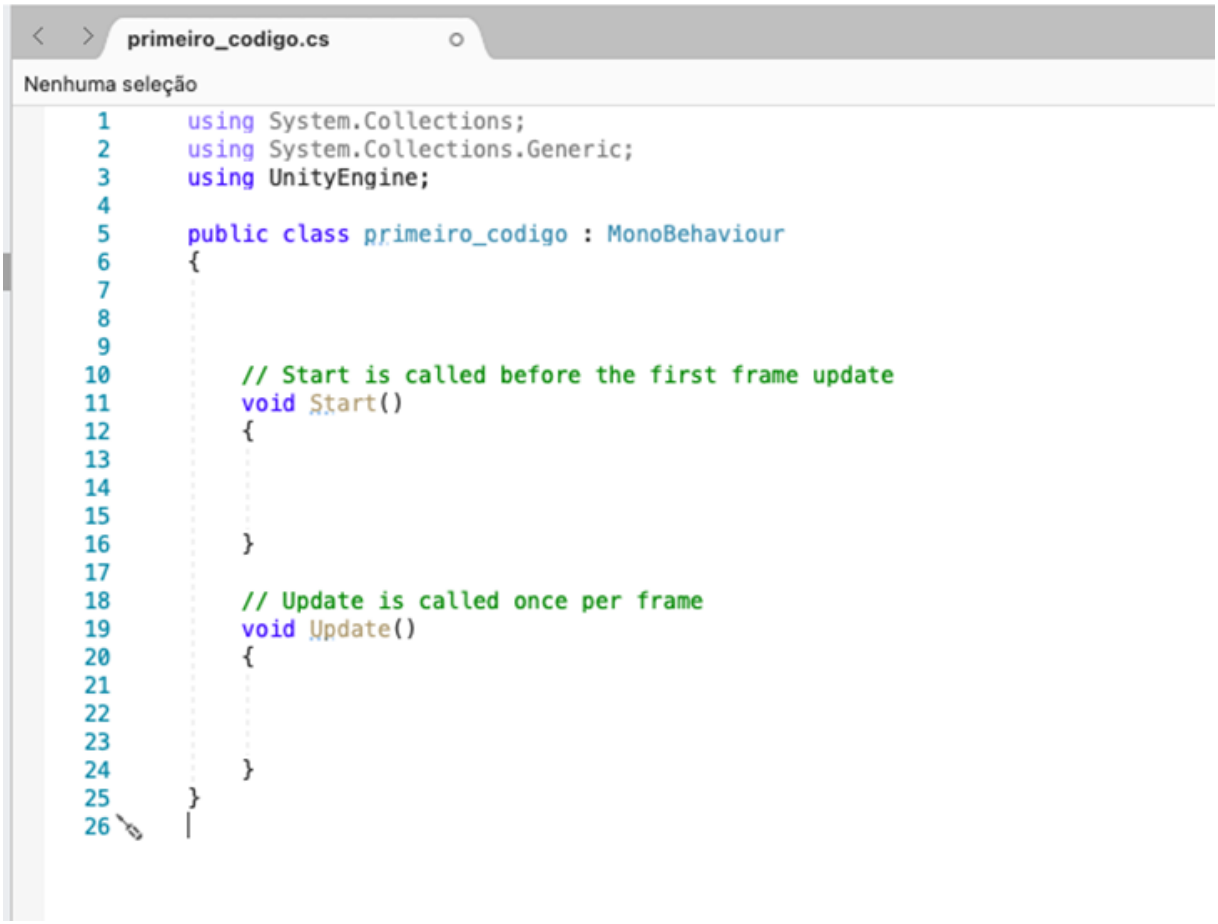

Imagem 2: Tela da Unity | Fonte: Autor (print de tela)

Vamos conhecer esta estrutura. Veja que, por estamos usando o visual studio, as classes, métodos, comandos e comentários, cada um tem uma cor identificando sua estrutura. Isso é muito bom para termos um código organizado e saber com que estrutura estamos trabalhando.

Vejam, que toda vez que criarmos um script em C#, ele vai criar esta estrutura da seguinte forma:

Void Start: O que se trata isso?

Aqui deve ser inserido tudo que irá iniciar no nosso game, ou seja, tudo que for escrito aqui, será executado uma única vez, no início do game.

Já no método **Void Update**, aqui deve ser inserido tudo que será atualizado no game. Podemos citar por exemplo, toda vez que o personagem for atingido ou atingir algum objeto do cenário, ou o personagem coletar algo que irá receber pontuação no nosso game, etc.

Outro ponto muito legal de se trabalhar com as ferramentas de desenvolvimento como o Visual Studio, é que, como ele já carrega todas as bibliotecas da programação C# e o próprio visual studio já vai identificando pra nós a *syntax* do comando ou do método e nos auxilia na formatação adequada do preenchimento completo do comando.

Vamos ver agora, como podemos declarar uma variável dentro do nosso Script em C#:

Imagem 3: Tela da Unity | Fonte: Autor (print de tela)

Veja que, devemos sempre iniciar a variável dentro da classe já criada (ou criar um classe específica), e antes do método Start.

Vamos aprender como trabalhar com operadores aritméticos na unity:

Vamos declarar duas variáveis do tipo inteiro (antes do método Start):

int valor1,valor2,totalsoma;

Veja que podemos declarar variáveis do mesmo tipo em uma única linha.

Agora, dentro do método Start, vamos realizar a operação aritmética de soma e solicitar que a unity exiba o resultado no painel console:

primeiro_codigo.cs

primeiro_codigo ▸ Start()

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
5   public class primeiro_codigo : MonoBehaviour
6   {
7
8       int valor1,valor2,totalsoma;
9
10      // Start is called before the first frame update
11      void Start()
12      {
13          valor1 = 50;
14          valor2 = 10;
15
16          totalsoma = valor1 + valor2;
17
18          print(totalsoma);
19
20      }
21
22      // Update is called once per frame
23      void Update()
24      {
25
26
27
28      }
29  }
30
```

Imagem 4: Tela da Unity | Fonte: Autor (print de tela)

Salve o Script e execute na unity para visualizar o resultado no painel console:

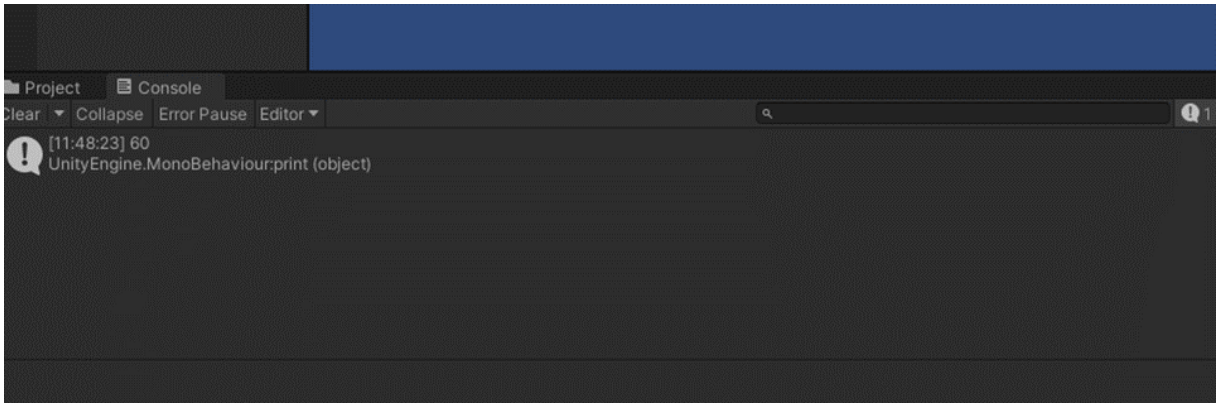

Imagem 5: Tela da Unity | Fonte: Autor (print de tela)

Neste módulo, vamos aprender bastante sobre variáveis na unity e como declarar.

Vamos conhecer a estrutura para declaração de variáveis dentro da unity através da linguagem C#.

int fichas = 100; (números inteiros)

float moedas = 10.5f; (números reais)

bool vitória = true; (booleana)

string nome_personagem = “astronauta”; (textos)

Exemplos de quando queremos realizar uma operação aritmética no C#:

```
totalsoma = valor1 + valor2;

totalsub = valor1 - valor2;

totalmult = valor1 * valor2;

totaldiv = valor1 / valor2;



print (totalsoma);

print (totalsub);

print (totalmult);

print (totaldiv);
```

Exemplos de como podemos trabalhar com condicional em C#:

```
if (fichaspersonagem >= qtdvida)

    {

       print(“personagem pode adquirir vida”);

    }

    else

    {
```

```
        print("Saldo insuficiente para adquirir vida");
    }
```

Atividade Extra

Acesse a o site https://docs.microsoft.com/pt-br/dotnet/csharp/ e conheça detalhamente todos os recursos do c#. acesse os menus e realize os testes online direto no site como funciona cada classe, métodos, comandos, etc.

Referência Bibliográfica

UNITY, Software. Site do Produto.

https://unity.com/pt/

https://docs.microsoft.com/pt-br/dotnet/csharp/

Ir para exercício